7p

Artificial Intelligence Project--RLE and MIT Computation Center
Symbol Manipulating Language--Memo 13--The Maling-Silver Read Program

K. Maling

Three types of expressions can read

(1) m - expressions

(2) s - expressions

(3) algebraic expressions.

The program uses RDB, which means that single embedded blanks may be part of a print name; that a left parenthesis followed by, or a right parenthesis preceded by, any combination of periods and commas is treated as a special parenthesis; and that any combination of + - = * 1 is an "operation" group.

It is written in LISP and when used with the apply operator, the cards bearing the expressions must follow the STOP card. The card immediately following the STOP card must contain the special list of operation and punctuation characters, many of which can only be read in by the use of RDB. A function DEFPRED uses this card to generate predicates for these characters.

RDA and RDB have independent buffers so that they can be used in the same program without confusion. Betern does for RDB what INTERN does for RDA. A Rollo expression becomes available to the apply operator by executing the function ARGLIST, which has no arguments.

Unless an expression is preceded by ++ to indicate "QUOTE" or +* to indicate algebraic mode; it will be assumed to be an in-expression. ARGLIST expects one or several self-contained expressions separated by commas and terminating with any symbol or character other than a left parenthesis, comma, period or operation. The value of ARGLIST is a list of translated expressions.

The operation and punctuation characters consist of the following set

==	define
--	yields
=	equal
+	or, plus
*	and, times
/	of, ratio
-	not, minus

** power
/* lambda
++ quote
-+ unquote
+* algebraic mode
-* m - mode
.   m - expression parenthesis, s - expression connective
(see below
)

The program is not debugged to the point of my happiness, and the following items may warrant attention.

(1) S - expressions:
Add implied comma for these combinations
) ( ) symbol group symbol group (
Introduce special parentheses.

(2) Algebraic expressions.
The very simple function "INFIX" does not make the proper association in certain special cases, namely
a+b**c*d is read as (a+(b**(c*d)))
instead of (a+((b**c)*d))
The more complex equivalent function for m - expressions solves this difficulty and the method can be copied for the algebraic mode.
If already in the algebraic mode, ** should be ignored. This is not so now.

(3) M - expressions.
If already in the m - mode, -* should be ignored. This is not so now.
The period - parenthesis should not require a matching period if the latter is at the beginning or end of the expression.
No use has been made of the fact that thevalue of RD is OPER Punct LPAR RPAR etc. This results in a slower program.
There is no special character for PROG.

Usage and Errors.

The function ARGLIST has as its value a list of expressions. That is why the examples given appear to have one redundant nesting of parentheses around the equivalent s - expressions. Suppose we punch on a card

1*(u)(u=++YES--Print/++OK,T--NIL),++YES $$

Then this card is placed after the cperation characters card which follows the STOP card.

Let the apply operator now execute

(EVAL,(ARGLIST),())

()

()

Then the above function will be applied to its argument.

The most common error is the failure to leave a space between punctuation characters, or operation characters which are intended to be distinct.

e.g. U=++YES instead of U= ++YES

or (A--B,.C+D.--E) instead of (A--B, .C+D.--E)

An expression or list of expressions separated by commas must have a termination symbol. $ $ $ causes the rest of the card to be ignored. This is a feature of RDB.

Once a grammatical error occurs the reading process usually gets out of step; it may therefore be advantageous to insert a dummy arglist with this particular termination symbol between expressions. This thought has not been debugged.

Expressions included in any one list may be of any kind in any order, in - s - or algebraic, but care must be taken to restore in - mode after an algebraic expression, if a non-algebraic expression follows within the same arglist. Every arglist starts out in the in-mode anew.

M-Expressions

The notation in which in-expressions are written is designed to minimize the number of parentheses used. There are two characters that are interpreted as or imply parentheses, namely the slash and the period.

CAR/L translates to ((CAR,L))

/READ translates to ((READ))

a*.b+c./d translates to ((AND,A,(AND,(OR,B,C),D)))

Thus functions of one or more arguments can be written without parentheses. For functions of more than one argument however, one writes

LIST (A,B,C,D) translates to ((LIST,A,B,C,D))

Binary connectives have been given an order of dominance. In order of decreasing strength they are

→ * / ; DEF, YIELDS, EQUAL, OR, AND, OF

Note that + and * are binary, and A*B*C*D is correctly (but not efficiently) translated.

This order of dominance implies the existence of parentheses.

A = B/C * D →E=F+G will be read as if

(A= (((B/C)*D) → (E=(F+G))))

A different method of parenthesizing can be imposed by inserting periods as if they were parentheses. There is no ambiguity since a period in front of a connective must be a closing parenthesis and a period after a connective must be an opening parenthesis. They must be matched.

Periods are usually necessary when NOT is intended, e.g. -.ATOM/L1. translates to ((NOT,(ATOM,L1)))

Normal parenthesis may enclose

1) the arguments of a function
2) the pairs of a conditional expression
3) the variables of a Lambda expression

and the terms are separated by commas. Normal parentheses should never be used in any other sense.

(P1→E1,P2→E2) translates to ((COND,(P1,E1,),(P2,E2)))

λ(X,Y)Z(U,V) translates to (((LAMBDA,(X,Y),Z),U,V))

λ(X,Y)(X→Y) translates to ((LAMBDA,(X,Y),(COND,(X,Y))))

Note that the triplet "λ(variables)expression" can be juxtaposed to a list of arguments just like the name of a function.

If one is in the algebraic mode and the first symbol of the next expression is -*, then the remainder of the expression will be read as an in-expression.

S-Expressions

The "quote" symbol is ++ and must be followed by a grammatically correct S-expression, including all commas. However, any number of trailing right parentheses can be made up by the "unquote" symbol which is -+ . If the number of right parentheses is correct, no unquote symbol is required to restore the in-expression mode. Periods are permitted in the notation, so that ++(A.(B)) and ++(A,B) are read identically. So also would be ++(A.(B-+ of course. Every s-expression which has been read into the computer is represented by a list whose first element is the object QUOTE. A null expression results from ++ -+ .

Algebraic Expressions

The operation characters are +* ** 1 - = where - can be used either as a unary or binary connective.

The order of dominance is = ± / * ** in decreasing strength. This implied method of parenthesizing can be modified with conventional parentheses.

A+B=C/D**E*-F translates to ((EQUAL,(PLUS,A,B),(RATIO,C,(TIMES,(POWER,D,E),(MINUS,F)))))

If the first character of an expression is +* then this expression will be translated as algebraic, and so will sunsequent ones until the mode is switched back to m-mode or the arglist meets a termination symbol.

The Conventions on RD(B)

1. Programming Conventions

The instruction TSX RD,+ at L causes reading from cards or type accordingly as sense switch 1 is down or up respectively. One or more results will be read by the input equipment until the termination of a group as defined below has been reached. If the instruction is not the first read instruction in a program then it may not activate input equipment. This occurs if both beginning and end of a group lie within one record that was read by a previous read instruction.

RD(B) has two exits. The normal return is to L+2, but an end of file condition or a triple dollar sign causes return to L+1. RD(B) has two values when the normal return takes place. In the decrement of the accumulator is the machine name of the group type (one of seven possibilities). The group itself is always stored in the same place, in a special list whose name is "VALVE".

2. Typographical Conventions

There are seven types of character groups

(a) Symbols
(b) Numbers
(c) Operations
(d) Punctuation
(e) Left parentheses
(f) Right parentheses
(g) Text

A symbol group always begins with a literal (AtoZ) and contains any of 26 literals, 10 digits or any number of single, isolated embedded blanks. Anything else terminates the group.

A number group always begins with a digit and contains any of 26 literals, 10 digits or one embedded period. The embedded period must not be adjacent to a literal. A blank may not be embedded. Anything else terminates the group.

There are six operation characters, + * / - - = . Hyphen and minus signs are treated without distinction. Any combination of operation characters or one alone forms an operation group. Also any expression which does not contain dollar signs may become part of an operation group by enclosing it by dollar signs and writing it immediately after the last character of

the operation group. The first enclosing dollar sign is read in as a blank, the second is ignored.

Any sequence of consecutive periods and/or commas is a punctuation group, provided it is not immediately preceded or followed by a left or right parenthesis respectively.

A left parenthesis group is a left parenthesis followed by any sequence of consecutive periods and/or commas.

A right parenthesis group is any sequence of consecutive periods and/or commas followed by a right parenthesis, providing this sequence is not preceded by a left parenthesis. If it is so preceded, the right parenthesis group consists of the right parenthesis alone.

Text is any sequence of characters (other than dollar signs) enclosed by a pair of dollar signs, providing the leading dollar sign is not preceded by an operation character. Upon read-in, the first dollar sign is replaced by a blank, and the second is ignored.

The number of characters in a group may be one to 120. The leading blank of text counts as one of the 120.

A double dollar sign terminating text, or a triple dollar sign at any time causes exit to L+2. A read instruction following such a termination will start by reading new record.

Blanks which precede or interpolate groups are supressed.

A record contains 72 characters, whether on cards or tape. These characters are punched into columns 1-72 of cards. The 72nd character of a record is followed without interruption by the first character of the next record. If the column punching does not represent one of the 48 legal characters then RD(B) goes to error. However, all combinations of legal characters are legal.

CS-TR Scanning Project
Document Control Form

Date : 11/30/95

Report # AIM-13

Each of the following should be identified by a checkmark:
Originating Department:

☒ Artificial Intellegence Laboratory (AI)
☐ Laboratory for Computer Science (LCS)

Document Type:

☐ Technical Report (TR) ☒ Technical Memo (TM)

☐ Other:_____

Document Information

Number of pages: 7(11-IMAGES)
Not to include DOD forms, printer intstructions, etc... original pages only.

Originals are:

☒ Single-sided or

☐ Double-sided

Intended to be printed as :

☒ Single-sided or

☐ Double-sided

Print type:

☐ Typewriter ☐ Offset Press ☐ Laser Print

☐ InkJet Printer ☐ Unknown ☒ Other: MIMEOGRAPH

Check each if included with document:

☐ DOD Form ☐ Funding Agent Form ☐ Cover Page

☐ Spine ☐ Printers Notes ☐ Photo negatives

☐ Other: _____

Page Data:

Blank Pages(by page number): _____

Photographs/Tonal Material (by page number): _____

Other (note description/page number):

Description :	Page Number:
IMAGE MAP: (1-7) 1-5, 2 APPENDIX PAGES	
(8-11) SCANCONTROL, TRGT'S (3)	

Scanning Agent Signoff:

Date Received: 11/30/95 Date Scanned: 12/14/95 Date Returned: 12/14/95

Scanning Agent Signature: Michael W. Cook

Scanning Agent Identification Target

Scanning of this document was supported in part by the **Corporation for National Research Initiatives,** using funds from the **Advanced Research Projects Agency** of the **United states Government** under Grant: **MDA972-92-J1029.**

The scanning agent for this project was the **Document Services** department of the **M.I.T Libraries.** Technical support for this project was also provided by the **M.I.T. Laboratory for Computer Sciences.**